Section 08: Solutions

1. Regular Expressions

(a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Solution:

$$0 \cup ((1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*)$$

(b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Solution:

$$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)^* 0)$$

(c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

Solution:

$$(01 \cup 001 \cup 1^*)^*(0 \cup 00 \cup \varepsilon)111(01 \cup 001 \cup 1^*)^*(0 \cup 00 \cup \varepsilon)$$

2. CFGs

(a) All binary strings that end in 00.
Solution:

$$\mathbf{S} \rightarrow 0\mathbf{S} \mid 1\mathbf{S} \mid 00$$

(b) All binary strings that contain at least three 1's.
Solution:

$$\begin{aligned} \mathbf{S} &\rightarrow \mathbf{TTT} \\ \mathbf{T} &\rightarrow 0\mathbf{T} \mid \mathbf{T}0 \mid 1\mathbf{T} \mid 1 \end{aligned}$$

(c) All binary strings with an equal number of 1's and 0's.
Solution:

$$\mathbf{S} \to 0\mathbf{S}1\mathbf{S} \mid 1\mathbf{S}0\mathbf{S} \mid \varepsilon$$

and

$$\mathbf{S} \to \mathbf{SS} \mid 0\mathbf{S}1 \mid 1\mathbf{S}0 \mid \varepsilon$$

both work. Note: The fact that all the strings generated have the property is easy to show (by induction) but the fact that one can generate all strings with the property is trickier. To argue this that each of these is grammars is enough one would need to consider how the difference between the # of 0's seen and the # of 1's seen occurs in prefixes of any string with the property.

3. Structural Induction

(a) Consider the following recursive definition of strings.

Basis Step: `""` is a string

Recursive Step: If X is a string and c is a character then $\mathsf{append}(c, X)$ is a string.

Recall the following recursive definition of the function len:

$$\begin{aligned} \mathsf{len}(\texttt{""}) &= 0 \\ \mathsf{len}(\mathsf{append}(c, X)) &= 1 + \mathsf{len}(X) \end{aligned}$$

Now, consider the following recursive definition:

$$\begin{aligned} \mathsf{double}(\texttt{""}) &= \texttt{""} \\ \mathsf{double}(\mathsf{append}(c, X)) &= \mathsf{append}(c, \mathsf{append}(c, \mathsf{double}(X))). \end{aligned}$$

Prove that for any string X, $\mathsf{len}(\mathsf{double}(X)) = 2\mathsf{len}(X)$.

Solution:

For a string X, let $\mathsf{P}(X)$ be "$\mathsf{len}(\mathsf{double}(X)) = 2\mathsf{len}(X)$". We prove $\mathsf{P}(X)$ for all strings X by structural induction on X.

Base Case (X = `""`): By definition, $\mathsf{len}(\mathsf{double}(\texttt{""})) = \mathsf{len}(\texttt{""}) = 0 = 2 \cdot 0 = 2\mathsf{len}(\texttt{""})$, so $\mathsf{P}(\texttt{""})$ holds

Inductive Hypothesis: Suppose $\mathsf{P}(X)$ holds for some arbitrary string X.

Inductive Step: $\boxed{\text{Goal: Show that } \mathsf{P}(\mathsf{append}(c, X)) \text{ holds for any character } c.}$

$$\begin{aligned} \mathsf{len}(\mathsf{double}(\mathsf{append}(c, X))) &= \mathsf{len}(\mathsf{append}(c, \mathsf{append}(c, \mathsf{double}(X)))) && \text{[By Definition of double]} \\ &= 1 + \mathsf{len}(\mathsf{append}(c, \mathsf{double}(X))) && \text{[By Definition of len]} \\ &= 1 + 1 + \mathsf{len}(\mathsf{double}(X)) && \text{[By Definition of len]} \\ &= 2 + 2\mathsf{len}(X) && \text{[By IH]} \\ &= 2(1 + \mathsf{len}(X)) && \text{[Algebra]} \\ &= 2(\mathsf{len}(\mathsf{append}(c, X))) && \text{[By Definition of len]} \end{aligned}$$

This proves $\mathsf{P}(\mathsf{append}(c, X))$.

Conclusion: $\mathsf{P}(X)$ holds for all strings X by structural induction.

(b) Consider the following definition of a (binary) **Tree**:

Basis Step: $\bullet$ is a **Tree**.

Recursive Step: If L is a **Tree** and R is a **Tree** then $\texttt{Tree}(\bullet, L, R)$ is a **Tree**.

The function $\textsf{leaves}$ returns the number of leaves of a **Tree**. It is defined as follows:

$$\begin{aligned}
\textsf{leaves}(\bullet) &= 1\\
\textsf{leaves}(\texttt{Tree}(\bullet, L, R)) &= \textsf{leaves}(L) + \textsf{leaves}(R)
\end{aligned}$$

Also, recall the definition of $\textsf{size}$ on trees:

$$\begin{aligned}
\textsf{size}(\bullet) &= 1\\
\textsf{size}(\texttt{Tree}(\bullet, L, R)) &= 1 + \textsf{size}(L) + \textsf{size}(R)
\end{aligned}$$

Prove that $\textsf{leaves}(T) \geq \textsf{size}(T)/2 + 1/2$ for all $\textsf{Trees}$ T.

Solution:

For a tree T, let $\textsf{P}$ be $\textsf{leaves}(T) \geq \textsf{size}(T)/2 + 1/2$. We prove $\textsf{P}$ for all trees T by structural induction on T.

Base Case (T = •): By definition of $\textsf{leaves}(\bullet)$, $\textsf{leaves}(\bullet) = 1$ and $\textsf{size}(\bullet) = 1$. So, $\textsf{leaves}(\bullet) = 1 \geq 1/2 + 1/2 = \textsf{size}(\bullet)/2 + 1/2$, so $\textsf{P}(\bullet)$ holds.

Inductive Hypothesis: Suppose $\textsf{P}(L)$ and $\textsf{P}(R)$ hold for some arbitrary trees L, R.

Inductive Step: $\boxed{\text{Goal: Show that } \textsf{P}(\texttt{Tree}(\bullet, L, R)) \text{ holds.}}$

$$\begin{aligned}
\textsf{leaves}(\texttt{Tree}(\bullet, L, R)) &= \textsf{leaves}(L) + \textsf{leaves}(R) && \text{[By Definition of leaves]}\\
&\geq (\textsf{size}(L)/2 + 1/2) + (\textsf{size}(R)/2 + 1/2) && \text{[By IH]}\\
&= (1/2 + \textsf{size}(L)/2 + \textsf{size}(R)/2) + 1/2 && \text{[By Algebra]}\\
&= \frac{1 + \textsf{size}(L) + \textsf{size}(R)}{2} + 1/2 && \text{[By Algebra]}\\
&= \textsf{size}(T)/2 + 1/2 && \text{[By Definition of size]}
\end{aligned}$$

This proves $\textsf{P}(\texttt{Tree}(\bullet, L, R))$.

Conclusion: Thus, $\textsf{P}(T)$ holds for all trees T by structural induction.

(c) Prove the previous claim using strong induction. Define $P(n)$ as "all trees T of size n satisfy $\textsf{leaves}(T) \geq \textsf{size}(T)/2 + 1/2$". You may use the following facts:

- For any tree T we have $\textsf{size}(T) \geq 1$.
- For any tree T, $\textsf{size}(T) = 1$ if and only if $T = \bullet$.

If we wanted to prove these claims, we could do so by structural induction.

Note, in the inductive step you should start by letting T be an arbitrary tree of size $k + 1$.

Solution:

Let $P(n)$ be "all trees T of size n satisfy $\textsf{leaves}(T) \geq \textsf{size}(T)/2 + 1/2$". We show $P(n)$ for all integers $n \geq 1$ by strong induction on n.

Base Case: Let T be an arbitrary tree of size 1. The only tree with size 1 is $\bullet$, so $T = \bullet$. By definition, $\textsf{leaves}(T) = \textsf{leaves}(\bullet) = 1$ and thus $\textsf{size}(T) = 1 = 1/2 + 1/2 = \textsf{size}(T)/2 + 1/2$. This shows the base case holds.

Inductive Hypothesis: Suppose that $P(j)$ holds for all integers $j = 1, 2, \ldots, k$ for some arbitrary integer $k \geq 1$.

Inductive Step: Let T be an arbitrary tree of size $k + 1$. Since $k + 1 > 1$, we must have $T \neq \bullet$. It follows from the definition of a tree that $T = \texttt{Tree}(\bullet, L, R)$ for some trees L and R. By definition, we have

$\text{size}(T) = 1 + \text{size}(L) + \text{size}(R)$. Since sizes are non-negative, this equation shows $\text{size}(T) > \text{size}(L)$ and $\text{size}(T) > \text{size}(R)$ meaning we can apply the inductive hypothesis. This says that $\text{leaves}(L) \geq \text{size}(L)/2 + 1/2$ and $\text{leaves}(R) \geq \text{size}(R)/2 + 1/2$.

We have,

$$\begin{aligned}
\text{leaves}(T) &= \text{leaves}(\texttt{Tree}(\bullet, L, R)) \\
&= \text{leaves}(L) + \text{leaves}(R) && \text{[By Definition of leaves]} \\
&\geq (\text{size}(L)/2 + 1/2) + (\text{size}(R)/2 + 1/2) && \text{[By IH]} \\
&= (1/2 + \text{size}(L)/2 + \text{size}(R)/2) + 1/2 && \text{[By Algebra]} \\
&= \frac{1 + \text{size}(L) + \text{size}(R)}{2} + 1/2 && \text{[By Algebra]} \\
&= \text{size}(T)/2 + 1/2 && \text{[By Definition of size]}
\end{aligned}$$

This shows $P(k+1)$.

Conclusion: $P(n)$ holds for all integers $n \geq 1$ by the principle of strong induction.

Note, this proves the claim for all trees because every tree T has some size $s \geq 1$. Then $P(s)$ says that all trees of size s satisfy the claim, including T.

4. Walk the Dawgs

Suppose a dog walker takes care of $n \geq 12$ dogs. The dog walker is not a strong person, and will walk dogs in groups of 3 or 7 at a time (every dog gets walked exactly once). Prove the dog walker can always split the n dogs into groups of 3 or 7.

Solution:

Let $P(n)$ be "a group with n dogs can be split into groups of 3 or 7 dogs." We will prove $P(n)$ for all natural numbers $n \geq 12$ by strong induction.

Base Cases $n = 12, 13, 14,$ **or** 15**:** $12 = 3 + 3 + 3 + 3$, $13 = 3 + 7 + 3$, $14 = 7 + 7$, So $P(12)$, $P(13)$, and $P(14)$ hold.

Inductive Hypothesis: Assume that $P(12), \ldots, P(k)$ hold for some arbitrary $k \geq 14$.

Inductive Step: $\boxed{\text{Goal: Show } k+1 \text{ dogs can be split into groups of size 3 or 7.}}$

We first form one group of 3 dogs. Then we can divide the remaining $k-2$ dogs into groups of 3 or 7 by the assumption $P(k-2)$. (Note that $k \geq 14$ and so $k-2 \geq 12$; thus, $P(k-2)$ is among our assumptions $P(12), \ldots, P(k)$.)

Conclusion: $P(n)$ holds for all integers $n \geq 12$ by by principle of strong induction.

5. Reversing a Binary Tree

Consider the following definition of a (binary) **Tree**.

Basis Step `Nil` is a **Tree**.

Recursive Step If L is a **Tree**, R is a **Tree**, and x is an integer, then $\texttt{Tree}(x, L, R)$ is a **Tree**.

The sum function returns the sum of all elements in a **Tree**.

$$\begin{aligned}
\text{sum}(\texttt{Nil}) &= 0 \\
\text{sum}(\texttt{Tree}(x, L, R)) &= x + \text{sum}(L) + \text{sum}(R)
\end{aligned}$$

The following recursively defined function produces the mirror image of a **Tree**.

$$\begin{aligned}
\mathsf{reverse}(\mathtt{Nil}) &= \mathtt{Nil} \\
\mathsf{reverse}(\mathtt{Tree}(x, L, R)) &= \mathtt{Tree}(x, \mathsf{reverse}(R), \mathsf{reverse}(L))
\end{aligned}$$

Show that, for all **Tree**s T that

$$\mathsf{sum}(T) = \mathsf{sum}(\mathsf{reverse}(T))$$

Solution:

For a **Tree** T, let $P(T)$ be "$\mathsf{sum}(T) = \mathsf{sum}(\mathsf{reverse}(T))$". We show $P(T)$ for all **Tree**s T by structural induction.

Base Case: By definition we have $\mathsf{reverse}(\mathtt{Nil}) = \mathtt{Nil}$. Applying sum to both sides we get $\mathsf{sum}(\mathtt{Nil}) = \mathsf{sum}(\mathsf{reverse}(\mathtt{Nil}))$, which is exactly $P(\mathtt{Nil})$, so the base case holds.

Inductive Hypothesis: Suppose $P(L)$ and $P(R)$ hold for some arbitrary **Tree**s L and R.

Inductive Step: Let x be an arbitrary integer. $\boxed{\text{Goal: Show } P(\mathtt{Tree}(x, L, R)) \text{ holds.}}$

We have,

$$\begin{aligned}
\mathsf{sum}(\mathsf{reverse}(\mathtt{Tree}(x, L, R))) &= \mathsf{sum}(\mathtt{Tree}(x, \mathsf{reverse}(R), \mathsf{reverse}(L))) && \text{[Definition of reverse]} \\
&= x + \mathsf{sum}(\mathsf{reverse}(R)) + \mathsf{sum}(\mathsf{reverse}(L)) && \text{[Definition of sum]} \\
&= x + \mathsf{sum}(R) + \mathsf{sum}(L) && \text{[Inductive Hypothesis]} \\
&= x + \mathsf{sum}(L) + \mathsf{sum}(R) && \text{[Commutativity]} \\
&= \mathsf{sum}(\mathtt{Tree}(x, L, R)) && \text{[Definition of sum]}
\end{aligned}$$

This shows $P(\mathtt{Tree}(x, L, R))$.

Conclusion: Therefore, $P(T)$ holds for all **Tree**s T by structural induction.